TRANSMITTAL OF APPEAL BRIEF (Large Entity)

Docket No.: MCT.0132US

In Re Application Of: **James McKeeth**

Application No.	Filing Date	Examiner	Customer No.	Group Art Unit	Confirmation No.
09/449,782	11-26-1999	Mary J. Steelman	21690	2191	6698

Invention: **Command Line Output Redirection**

COMMISSIONER FOR PATENTS:

Transmitted herewith ~~in triplicate~~ is the Appeal Brief in this application, with respect to the Notice of Appeal filed on **September 27, 2005.**

The fee for filing this Appeal Brief is: **$500.00**

☒ A check in the amount of the fee is enclosed.

☐ The Director has already been authorized to charge fees in this application to a Deposit Account.

☒ The Director is hereby authorized to charge any fees which may be required, or credit any overpayment to Deposit Account No. **20-1504** (MCT.0132US)

☐ Payment by credit card. Form PTO-2038 is attached.

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

Signature

Dated: **November 22, 2005**

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Suite 100
Houston, TX 77024
Telephone: (713) 468-8880, ext. 304
Facsimile: (713) 468-8883

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to "Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450" [37 CFR 1.8(a)] on **11-22-2005**.
(Date)

Signature of Person Mailing Correspondence

Ginger Yount

Typed or Printed Name of Person Mailing Correspondence

CC:

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants:	James McKeeth	§	Art Unit:	2191
		§		
Serial No.:	09/449,782	§		
		§	Examiner:	Mary J. Steelman
Filed:	November 26, 1999	§		
		§		
For:	Command Line Output Redirection	§	Atty. Dkt. No.:	MCT.0132US (MUEI-0531.00/US)

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF PURSUANT TO 37 C.F.R § 41.37

Sir:

The final rejection of claims 1-21 and 23-25 is hereby appealed.

I. REAL PARTY IN INTEREST

The real party in interest is Micron Technology, Inc.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF THE CLAIMS

Claims 1-21 and 23-25 have been finally rejected and are the subject of this appeal.

Claim 22 has been cancelled.

Date of Deposit: November 22, 2005

I hereby certify under 37 CFR 1.8(a) that this correspondence is being deposited with the United States Postal Service as **first class mail** with sufficient postage on the date indicated above and is addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313.

Ginger Yount

IV. STATUS OF AMENDMENTS

The claims have not been amended after final rejection.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in each of the independent claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element.

Independent claim 1 recites a method comprising:

> invoking, by an application, a call of a command line utility, the application providing an identifier in the call of the command line utility (Specification, p. 4, lines 5-22; Fig. 2:202; Fig. 3:302);
>
> receiving output from the command line utility (Fig. 2:204; Fig. 3:304; Specification, p. 3, lines 9-12, p. 6, lines 6-9);
>
> storing the command line utility output in a system storage at a location identified by the identifier (Fig. 2:206; Fig. 3:306; Specification, p. 3, lines 11-14; p. 6, lines 9-16); and
>
> retrieving, by the application, the command line utility output from the system storage at the location identified by the identifier (Specification, p. 3, lines 18-20; p. 6, lines 22-25).

Independent claim 15 recites a program storage device (Fig. 4:404), readable by a computer (Fig. 4:400), comprising instructions stored on the program storage device for causing the computer to:

cause an application to invoke a call of a command line utility, the application providing an identifier in the call of the command utility (Specification, p. 4, lines 5-22; Fig. 2:202; Fig. 3:302);

receive output from the command line utility (Fig. 2:204; Fig. 3:304; Specification, p. 3, lines 9-12, p. 6, lines 6-9);

store the command line utility output in system storage at a location identified by the identifier (Fig. 2:206; Fig. 3:306; Specification, p. 3, lines 11-14; p. 6, lines 9-16); and

cause the application to retrieve the command line utility output from the storage at the location identified by the identifier (Specification, p. 3, lines 18-20; p. 6, lines 22-25).

Independent claim 21 recites a computer system (Fig. 4:400), comprising:

a processor (Fig. 4:406);

a command line utility (Specification, p. 4, lines 5-19);

an application executable on the processor, the application to call the command line utility, the application to provide an identifier in the call (Specification, p. 4, lines 5-22; Fig. 2:202; Fig. 3:302);

a system storage (Fig. 4:404) having a location identified by the identifier, the location identified by the identifier to store an output of the command line utility (Specification, p. 3, lines 11-14; p. 6, lines 9-16; Fig. 2:206; Fig. 3:306),

the application to retrieve the command line utility output from the location identified by the identifier (Specification, p. 3, lines 18-20; p. 6, lines 22-25).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Claims 1-21 And 23-25 Were Rejected Under 35 U.S.C. § 103 Over U.S. Patent No. 6,182,279 (Buxton) In View Of Brian Livingston, "Windows 95 Secrets," 3rd Ed. (Livingston).

VII. ARGUMENT

The claims do not stand or fall together. Instead, Appellant presents separate arguments for various independent and dependent claims. Each of these arguments is separately argued below and presented with separate headings and sub-headings as required by 37 C.F.R. § 41.37(c)(1)(vii).

1. Claims 1-8, 10, and 15-21.

It is respectfully submitted that a *prima facie* case of obviousness has not been established with respect to claim 1 over the asserted combination of Buxton and Livingston for at least the following two reasons: (1) no motivation or suggestion existed to combine the teachings of Buxton and Livingston; and (2) even if combined, the hypothetical combination of Buxton and Livingston fails to teach or suggest *all* elements of claim 1. *See* M.P.E.P. § 2143 (8th ed., Rev. 3), at 2100-135.

Claim 1 recites a method that comprises:

- invoking, by an application, a call of a *command line utility*, the application *providing an identifier* in the call of the command line utility;
- receiving output from the command line utility;
- storing the *command line utility output* in a system storage at a location identified by the *identifier*; and
- retrieving, by the application, the *command line utility output* from the system storage at the location identified by the *identifier*.

The Examiner cited column 7, line 65 through column 8, line 10 as disclosing the task (recited in claim 1) of invoking, by an application, a call of the *command line utility*, the application providing an identifier in the call of the command line utility. 6/24/2005 Office Action at 2-3. Note that the Examiner asserted that this element of claim 1 is taught by Buxton

despite the concession made by the Examiner that Buxton fails to disclose a "command line utility." *Id.* at 4.

The cited passage of Buxton refers to an Object Linking and Embedding (OLE) container 220 that interacts with a WIN 32 application programming interface (API) 240 through OLE libraries 230 to insert OLE objects or controls into an operating system registry 250. Buxton, 7:66-8:2. Examples of the OLE container that can interact with the WIN 32 API 240 to insert OLE objects or controls into the operating system registry consist of Lotus Notes and Microsoft Word. Buxton, 8:2-6. As further explained in the cited passage, OLE libraries 230 include a set of system-level services in accordance with the OLE specification that function to call the WIN 32 API to locate registry objects. Buxton, 8:6-11. There is no teaching or suggestion anywhere in Buxton that the OLE container 220 (which can modify registry entries) can be substituted with a command line utility as recited in claim 1.

The Examiner relied upon Livingston as teaching the proposed modification of Buxton to achieve the claimed invention. Specifically, the Examiner relied upon the teaching in Livingston regarding the Registry Editor, which as taught by Livingston, allows a user to go to a DOS prompt (without starting Windows) to enable the editing of the registry. 6/24/2005 Office Action at 4. The Registry Editor described in Livingston is started manually by a user, with the user entering a command to start editing the registry. There is no suggestion in Livingston or in Buxton that the DOS Registry Editor taught by Livingston can be substituted for the OLE container described in the cited passage of Buxton relied upon by the Examiner. In fact, a person of ordinary skill in the art would not have been motivated to make the proposed modification, since there existed no suggestion anywhere that it would even be desirable to incorporate the DOS Registry Editor of Livingston into Buxton. *See In re Fritch*, 972 F.2d 1260, 1266,

23 U.S.P.Q.2d 1780 (Fed. Cir. 1992) ("The mere fact that the prior art may be modified in the manner suggested by the Examiner does not make the modification obvious unless the prior art suggested the *desirability* of the modification.") (emphasis added). The OLE container of Buxton has to interact with WIN32 APIs and OLE libraries to modify registry entries. Clearly, a person of ordinary skill in the art would not have substituted the DOS Registry Editor of Livingston for the OLE container of Buxton as doing so would clearly defeat the intended purpose of Buxton – namely, to use OLE components to enable modification of registry entries. Thus, clearly, the prior art does not suggest the desirability of the modification of Buxton proposed by the Examiner.

What the Examiner has engaged in is a classic example of impermissible hindsight reconstruction that picks and chooses un-related elements from prior art references to achieve the claimed invention, where no motivation or suggestion existed to make the proposed combination. *See In re Fine*, 837 F.2d 1071, 1075, 5 U.S.P.Q.2d 1596 (Fed. Cir. 1988) ("One cannot use hindsight reconstruction to pick and choose among isolated disclosures in the prior art to deprecate the claimed invention.").

In the Advisory Action dated September 13, 2005, the Examiner stated that "Buxton suggested receiving commands via command line, which results in modifying the registry (system storage) and storage." Advisory Action at 2. The reference to the command line made by the Examiner apparently refers to a passage in column 8, lines 45-52, of Buxton. This passage discloses that a user can use a user interface to interact with component system 200 of Buxton (see Fig. 2), where the user interface may be implemented with a simple command line interpreter or may have a more sophisticated graphic user interface. This command line interpreter discussed in Buxton refers to a user interface to allow the user to invoke components

of the component system 200 in Buxton. However, this command line interpreter of Buxton clearly does not suggest the command line utility of claim 1, which is invoked by an application, and which enables output to be received from this command line utility, which output is stored in a system storage at a location identified by the identifier provided by the application in the call of the command line utility. Therefore, the reference to a command line in a user interface by Buxton does not provide any suggestion of modifying the OLE container of Buxton with the registry editor of Livingston.

The Examiner has failed to explain why it would be desirable to incorporate the Registry Editor taught by Livingston in place of the OLE container 220 of Buxton. The OLE container 220 has to interact with WIN32 APIs 240 through OLE libraries 230 for inserting OLE objects or controls into an operating system registry 250. Buxton, 7:66-8:2. In fact, inserting the DOS Registry Editor of Livingston in place of the OLE container 220 of Buxton would render the Buxton system inoperative for its intended purpose, since the DOS Registry Editor would *not* be able to interact with the WIN32 APIs 240 through OLE libraries 230 to insert OLE objects or controls into the operating system registry.

The various dictionary definitions cited by the Examiner in the Advisory Action do not provide any suggestion of modifying Buxton based on the teachings of Livingston. Therefore, even if the final rejection is supplemented with the additional material cited by the Examiner in the Advisory Action, the Examiner has still failed to establish a *prima facie* case of obviousness with respect to claim 1.

The Examiner also makes another incorrect assertion on page 14 of the 6/24/2005 Office Action. There, the Examiner stated that "Buxton's invention inherently calls the registry editor (col. 8, line 10) to 'insert and create object dialog ...' when registering custom components in the

registry." 6/24/2005 Office Action at 14. It appears that the Examiner is asserting that Buxton inherently calls the Registry Editor of Livingston, which is a DOS Registry Editor. This statement is directly inconsistent with the teaching of Buxton which states that the OLE container 220 interacts with WIN32 APIs 240. There is absolutely no indication that the DOS Registry Editor of Livingston can interact with WIN32 APIs – therefore, the statement that "Buxton's invention inherently calls the registry editor" of Livingston is erroneous.

The remaining elements of claim 1 recite various tasks performed with respect to the command line utility, including receiving output from the command line utility, storing the command line utility output in a system storage at the location identified by the identifier, and retrieving, by the application, the command line utility output from the storage location at the location identified by the identifier. Note that the application that retrieves the command line utility output from the system storage at the location identified by the identifier is the *same* application that invokes the call of the command line utility, and the *same* application that provides the identifier in the call of the command line utility. The Examiner identified OLE libraries as being the application that retrieves output from a system storage. 6/24/2005 Office Action at 3.

The OLE libraries cannot be the application of claim 1. As discussed above, the Examiner has also identified the OLE libraries 230 as being the utility that is invoked by an application. (Note the citation of column 8, line 7, of Buxton by the Examiner which points specifically to the system-level services that make up the OLE libraries 230.) By also equating the OLE libraries with the application of claim 1, the Examiner appears to be indicating that the OLE libraries can call itself (note that claim 1 recites that the application invokes a call of the command line utility). However, this reading is clearly erroneous, as Buxton is clear in teaching that the OLE

libraries 230 do not call itself. In fact, it is the OLE container 220 (separate from the OLE libraries 230) that interacts with APIs 240 through the OLE libraries 230 to modify the operating system registry. The internal inconsistency of the rejection is another reason that the *prima facie* case is defective.

A further basis that the *prima facie* case is defective is that the hypothetical combination of Buxton and Livingston clearly fails to teach *all* elements of the claim. As explained above, Buxton clearly fails to teach invoking, by an application, a call of a *command line utility*, where the application provides an identifier in the call of the command line utility. Livingston also fails to disclose or suggest this element that is missing from Buxton, since the Registry Editor of Livingston is not invoked by an application that also provides an identifier in the call of the Registry Editor. Therefore, because neither Buxton nor Livingston teaches or suggests all elements of claim 1, the hypothetical combination of Buxton and Livingston clearly does not disclose or suggest all elements of the claim.

In response, the Examiner in the Advisory Action cited column 14, lines 20-28, of Buxton. This passage of Buxton refers to creation of a template by a template builder, and the inclusion of a registration key or subkey with the template 420. The registration key or subkey is used to identify a template 420, not to identify a location for storing a command line utility output as recited in claim 1. If the Examiner is asserting that the template builder of Buxton is a command line utility, that assertion is clearly inconsistent with an explicit finding by the Board of Patent Appeals and Interferences that the template builder is *not* a command line utility, and thus any output of the template builder cannot be considered the output of the command line utility. *See* Decision on Appeal, dated September 24, 2004, at 4. Therefore, the identifier of the

template in Buxton has nothing to do with the identifier of a location of a command line utility output.

For the foregoing reasons, it is respectfully requested that the final rejection of the above claims be reversed.

2. Claim 9.

Claim 9 depends indirectly from claim 1 and is allowable for at least the same reasons as for claim 1.

Moreover, with respect to dependent claim 9, the hypothetical combination of Buxton and Livingston does not teach or suggest that providing the identifier indicating a shared system memory identifies a system *clipboard memory*. The Examiner referred to column 11, line 6, of Buxton as teaching this element. Note that the cited passage of Buxton refers to an OLE data structure that acts as a generalized clipboard format. There is absolutely no suggestion of providing an identifier of a system clipboard memory, where such identifier identifies a location to *store command line utility output*.

For the foregoing reasons, it is respectfully requested that the final rejection of the above claim be reversed.

3. Claim 11.

Claim 11 depends from claim 1 and is thus allowable for at least the same reasons as for claim 1.

Moreover, there is no teaching or suggestion in the hypothetical combination Buxton and Livingston of receiving output from a command line utility through a subsequent command line output routine, as recited in claim 11. The Examiner cited column 8, lines 28-29, of Buxton as

teaching this feature. The cited passage refers to data items within a registry being retrievable by calls to WIN32 APIs. Retrieving data items from a registry through API calls is completely different from receiving output from a command line utility through a subsequent command line output routine.

For the foregoing reason, it is respectfully requested that the final rejection of the above claim be reversed.

4. Claims 12-14.

Claim 12 depends from claim 1 and is allowable for at least the same reasons as for claim 1.

Moreover, there is no suggestion in the hypothetical combination of Buxton and Livingston of associating each line of command line utility output with a line identifier in the system storage, as recited in claim 12. The Examiner cited column 3, lines 1-9, and column 13, lines 35-44, as teaching this feature. The cited passages refer to storing templates using key information, and storing each template in an ISTORAGE using a name that may be a decimal number. Neither the key nor the decimal number referred to in these passages constitute the line identifier of a command line utility output.

For the foregoing reason, it is respectfully requested that the final rejection of the above claims be reversed.

5. Claims 23-25.

Claims 23-25 depend from independent claims 1, 15, and 21, respectively, and are allowable for at least the same reasons as corresponding independent claims.

With respect to claim 23, the hypothetical combination of Buxton and Livingston fails to teach or suggest invoking a call to pipe output of a second command line utility to a first command line utility. The Examiner cited column 8, lines 6-7, and column 20, lines 17-43, of Buxton as teaching this feature. The cited passages referred to system-level services provided by the OLE libraries 230, and using a sub-key to read from a registry, determining whether a certificate exists by reading an appropriate registry entry, and if a certificate exists, determining whether or not a license period has expired. There is absolutely no suggestion whatsoever in the cited passages of Buxton of invoking a call to pipe output of a second command line utility to a first command line utility.

Claims 24 and 25 are allowable over the asserted combination of Buxton and Livingston for similar reasons as for claim 23.

For the foregoing reasons, it is respectfully requested that the final rejection of the above claims be reversed.

VIII. CONCLUSION

In view of the foregoing, reversal of all final rejections and allowance of all pending claims is respectfully requested.

Respectfully submitted,

Date: Nov. 22, 2005

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Suite 100
Houston, TX 77024
Telephone: (713) 468-8880
Facsimile: (713) 468-8883

APPENDIX OF APPEALED CLAIMS

The claims on appeal are:

1. A method comprising:

invoking, by an application, a call of a command line utility, the application providing an identifier in the call of the command line utility;

receiving output from the command line utility;

storing the command line utility output in a system storage at a location identified by the identifier; and

retrieving, by the application, the command line utility output from the system storage at the location identified by the identifier.

2. The method of claim 1, wherein providing the identifier comprises providing an identifier that identifies one or more entries in a system registry database.

3. The method of claim 2, wherein providing the identifier comprises providing a root key identifier.

4. The method of claim 3, wherein providing the root key identifier comprises providing a sub-key identifier.

5. The method of claim 2, wherein the system registry database comprises an operating system registry database.

6. The method of claim 1, wherein providing the identifier comprises providing a system storage identifier.

7. The method of claim 6, wherein providing the system storage identifier comprises providing an identifier indicating a system registry.

8. The method of claim 6, wherein providing the system storage identifier comprises providing an identifier indicating shared system memory.

9. The method of claim 8, wherein providing the identifier indicating shared system memory identifies a system clipboard memory.

10. The method of claim 1, wherein the act of receiving output from a command line utility comprises receiving output directly from the command line output utility.

11. The method of claim 1, wherein the act of receiving output from a command line utility comprises receiving output from the command line output utility through a subsequent command line output routine.

12. The method of claim 1, wherein the act of storing comprises associating each line of command line utility output with a line identifier in the system storage.

13. The method of claim 12, further comprising setting each line identifier to a value corresponding to a position of that line in the command line utility output.

14. The method of claim 12, further comprising setting a default value of the provided identifier to equal the total number of command utility output lines stored in the system storage.

15. A program storage device, readable by a computer, comprising instructions stored on the program storage device for causing the computer to:

cause an application to invoke a call of a command line utility, the application providing an identifier in the call of the command utility;

receive output from the command line utility;

store the command line utility output in system storage at a location identified by the identifier; and

cause the application to retrieve the command line utility output from the storage at the location identified by the identifier.

16. The program storage device of claim 15 wherein the instructions to store comprise instructions to store command line utility output in an operating system registry database.

17. The program storage device of claim 15 wherein the instructions to store comprise instructions to store command line utility output in an operating system maintained volatile memory.

18. The program storage device of claim 15 wherein the instructions to receive output comprise instructions to receive one or more lines of output from the command line utility, and the instructions to store further comprise instructions to store each of said one or more lines of output in the system storage.

19. The program storage device of claim 18 wherein the instructions to store further comprise instructions to associate a unique identifier with each of the one or more lines of output stored in the system storage.

20. The program storage device of claim 18 wherein the instructions to store further comprise instructions to set a value associated with the received identifier in the system storage equal to the number of lines of output stored in the system storage.

21. A computer system, comprising:

a processor;

a command line utility;

an application executable on the processor, the application to call the command line utility, the application to provide an identifier in the call;

a system storage having a location identified by the identifier, the location identified by the identifier to store an output of the command line utility,

the application to retrieve the command line utility output from the location identified by the identifier.

23. The method of claim 1, wherein the command line utility comprises a first command line utility, and wherein invoking the call by the application comprises invoking a call to pipe output of a second command line utility to the first command line utility,

wherein storing the command line utility output comprises storing the command line utility output of the first command line utility.

24. The program storage device of claim 15, wherein the command line utility comprises a first command line utility, and wherein invoking the call by the application comprises invoking a call to pipe output of a second command line utility to the first command line utility,

wherein storing the command line utility output comprises storing the command line utility output of the first command line utility.

25. The computer system of claim 21, wherein the command line utility comprises a first command line utility, the system further comprising a second command line utility, the application to invoke a call that causes output of the second command line utility to be piped to the first command line utility,

the location identified by the identifier to store output of the first command line utility.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.